Computer Organization and Architecture

Short notes for B.Tech CSE / IT · BCA · MCA · GATE CS

Uncodemy Handwritten Notes · free for students

I. Basic Structure of a Computer

- **Architecture** = what the programmer sees (instruction set, addressing modes, data types).
- **Organization** = how it is built (control signals, memory technology, buses).
- Main units: CPU (ALU + Control Unit + registers), Memory, I/O, connected by buses.

➤ Von Neumann vs Harvard

Von Neumann	Harvard
One memory for program and data	Separate memories
One bus, so a bottleneck	Separate buses, parallel fetch
General computers	DSPs, microcontrollers, caches (split L1)

➤ Important registers

Register	Job
PC (Program Counter)	Address of the next instruction
IR (Instruction Register)	Current instruction
MAR	Address to read / write in memory
MDR (MBR)	Data read from / to be written to memory
AC (Accumulator)	Holds ALU result

➤ Buses

- **Address bus** (one-way, CPU to memory), **data bus** (two-way), **control bus**.
- With k address lines, memory addressable = 2^k locations.

2. Instruction Set and Addressing Modes

- Instruction = **opcode** + **operand(s)**.
- Instruction cycle: fetch → decode → fetch operand → execute → store.
- Fetch micro-operations: MAR ← PC, MDR ← M[MAR], PC ← PC + 1, IR ← MDR.

➤ Instruction formats

Type	Example	Note
3-address	ADD R1, R2, R3	Short programs, long instructions
2-address	ADD R1, R2	Common (x86)
1-address	ADD X	Uses accumulator
0-address	ADD	Stack machine

➤ Addressing modes

Mode	Effective address (EA)	Use
Immediate	Operand is in the instruction	Constants
Direct	EA = address field	Global variables
Indirect	EA = M[address field]	Pointers
Register	Operand in register	Fast
Register indirect	EA = content of register	Pointers
Indexed	EA = address + index register	Arrays
Base register	EA = base register + offset	Relocation
PC-relative	EA = PC + offset	Branches
Auto-increment	EA = R, then R = R + 1	Loops over arrays

EXAM POINT Immediate needs 0 memory references for the operand, direct needs 1, indirect needs 2.

FORMULA If an instruction is w bits with opcode k bits, number of possible opcodes = 2^k. For expanding opcodes, count the unused combinations at each level.

RISC vs CISC

RISC	CISC
Few, simple instructions	Many, complex instructions
Fixed-length instructions	Variable length
Mostly 1 cycle per instruction	Many cycles per instruction
Load/store architecture	Memory operands in many instructions
Hardwired control	Often microprogrammed
ARM, RISC-V, MIPS	x86

3. Control Unit

Hardwired	Microprogrammed
Control signals from logic circuits	Control signals from microinstructions in control memory
Fast	Slower
Hard to change	Easy to change / add instructions
Used in RISC	Used in CISC

- **Horizontal microprogramming**: one bit per control signal, long words, more parallel.
- **Vertical microprogramming**: signals are encoded, short words, needs decoders.

FORMULA Control word with n signals in horizontal form = n bits. In vertical form with groups, bits = sum of $\lceil \log_2(\text{signals in group} + 1) \rceil$.

4. Computer Arithmetic

- Signed integers use 2's complement (see Digital Logic notes).
- **Booth's algorithm** multiplies signed numbers. Look at bit pair (Q0, Q−1): 10 → subtract M, 01 → add M, 00 / 11 → only shift.
- **Restoring and non-restoring division** for unsigned binary division.

IEEE 754 floating point

Format	Sign	Exponent	Mantissa	Bias
Single (32-bit)	1	8	23	127
Double (64-bit)	1	11	52	1023

FORMULA Value = (−1)^S x 1.M x 2^(E − bias), for normalised numbers.

EXAMPLE Represent −6.5 in single precision. 6.5 = 110.1 = 1.101 x 2^2. S = 1, E = 2 + 127 = 129 = 10000001, M = 101000...0. Bits: 1 10000001 10100000000000000000000 = hex C0D00000.

- Exponent all 0s: zero / denormal. Exponent all 1s: infinity (M = 0) or NaN (M ≠ 0).

5. Memory Hierarchy

- Registers → Cache → Main memory (RAM) → SSD / Disk. Going down: bigger, cheaper, slower.
- **Locality of reference**: temporal (same item used again soon) and spatial (nearby items used soon).

➤ Cache basics

- **Hit ratio** h = hits / total accesses.

FORMULA Average access time (simultaneous) = h·Tc + (1 − h)·Tm. Hierarchical (look in cache first) = h·Tc + (1 − h)(Tc + Tm).

EXAMPLE Tc = 10 ns, Tm = 100 ns, h = 0.9, hierarchical: 0.9 x 10 + 0.1 x (10 + 100) = 9 + 11 = 20 ns.

➤ Cache mapping

Mapping	Idea	Address split
Direct	Each block goes to exactly one line	Tag, Line, Offset
Fully associative	Block can go to any line	Tag, Offset
k-way set associative	Block goes to any line of one set	Tag, Set, Offset

- Direct mapping: line = block number mod number of lines. Simple but more conflict misses.

FORMULA Offset bits = log2(block size). Line bits = log2(number of lines). Set bits = log2(lines / k). Tag bits = address bits − other bits.

Solved example (direct mapped)

- Main memory 4 GB (32-bit address), cache 64 KB, block 32 B.
- Offset = log2 32 = 5 bits. Lines = 64 KB / 32 B = 2048, so line = 11 bits.
- Tag = 32 − 11 − 5 = 16 bits.
- If the same cache is 4-way: sets = 2048 / 4 = 512, set bits = 9, tag = 32 − 9 − 5 = 18 bits.

Replacement and write policies

- Replacement: LRU, FIFO, random (not needed in direct mapping).
- **Write-through**: write to cache and memory together. Simple, more memory traffic.
- **Write-back**: write to memory only when the block is replaced. Uses a dirty bit.
- Misses: **compulsory** (first access), **capacity** (cache too small), **conflict** (mapping clash).

6. Pipelining

- Split instruction execution into stages, so many instructions overlap. Classic 5 stages: IF, ID, EX, MEM, WB.

FORMULA Time for n instructions on k-stage pipeline = (k + n − 1) x cycle time. Speedup = (n·k·tn) / ((k + n − 1)·tp). Ideal max speedup = k.

EXAMPLE k = 5 stages, n = 100 instructions, cycle 2 ns. Pipelined time = (5 + 99) x 2 = 208 ns. Non-pipelined (each 10 ns) = 1000 ns. Speedup = 1000 / 208 = 4.81.

- Cycle time of pipeline = max stage delay + latch delay.

Hazards

Hazard	Cause	Fix
Structural	Two instructions need the same hardware	More hardware, split caches
Data (RAW, WAR, WAW)	Instruction needs result of an earlier one	Forwarding (bypassing), stalls
Control (branch)	Next instruction not known until branch resolves	Branch prediction, delayed branch

- In a simple in-order 5-stage pipeline only RAW hazards cause stalls; WAR and WAW matter in out-of-order pipelines.

FORMULA Effective CPI with stalls = 1 + (stall cycles per instruction). Example: 20% branches, 2 stall cycles each → CPI = 1 + 0.2 x 2 = 1.4.

7. I/O Organization

Method	Idea	CPU busy?
Programmed I/O	CPU keeps checking device status (polling)	Yes, wastes time
Interrupt-driven	Device interrupts CPU when ready	Only during transfer
DMA	DMA controller moves block directly to memory	Only at start and end

- **DMA modes**: burst (whole block at once), cycle stealing (one word at a time), transparent (only when the bus is free).
- **Interrupt**: CPU finishes current instruction, saves PC and state, runs the ISR, then returns.
- **Vectored interrupt**: device gives the ISR address. Priority by daisy chaining or a priority controller.

FORMULA Percentage of time CPU is blocked in cycle-stealing DMA = (time to transfer one word) / (time to prepare one word) x 100.

8. Performance Measures

FORMULA CPU time = Instruction count x CPI x clock cycle time = (IC x CPI) / clock rate.

FORMULA Amdahl's law: Speedup = 1 / ((1 − f) + f / s), where f = fraction improved, s = speedup of that part.

EXAMPLE 40% of a program is made 4 times faster. Speedup = 1 / (0.6 + 0.4/4) = 1 / 0.7 = 1.43.

- MIPS = clock rate / (CPI x 10^6).

9. Most Asked Questions (practice)

1. Explain the instruction cycle with micro-operations.
2. List addressing modes with examples and EA calculation.
3. RISC vs CISC. Hardwired vs microprogrammed control.
4. Multiply two signed numbers using Booth's algorithm.
5. Represent a decimal number in IEEE 754 single precision.
6. Find tag, set and offset bits for direct, associative and set-associative caches.
7. Calculate average memory access time for a two-level hierarchy.
8. Find speedup and time of a k-stage pipeline. Explain hazards and forwarding.
9. Compare programmed I/O, interrupt-driven I/O and DMA.
10. Apply Amdahl's law to a given improvement.

TIP GATE COA numericals are mostly cache address splitting, AMAT, pipeline speedup with stalls, IEEE 754 and addressing modes. Write the address split as a box every time.